

2013年度 卒業論文

アンドロイド搭載センサを利用した ながら走りの判別法

指導教員：渡辺 大地 講師
三上 浩司 准教授

メディア学部 ゲームサイエンスプロジェクト
学籍番号 M0110495
小澤 聡太郎

2013年度 卒業論文概要

論文題目

アンドロイド搭載センサを利用した
ながら走りの判別法

メディア学部

学籍番号：M0110495

氏名

小澤 聡太郎

指導
教員

渡辺 大地 講師
三上 浩司 准教授

キーワード

アンドロイド、ながら走り、センサ、GPS、自転車

近年、スマートフォンの普及によりユーザが増加している一方でマナーの悪いユーザによるスマートフォンを使用しながら歩く「ながら歩き」や自転車に乗りながら使用する「ながら走り」による事故が問題となっている。先行研究としてその「ながら歩き」を問題視し、判別する研究はいくつか存在する。しかし自転車に乗りながらスマートフォンを利用する「ながら走り」は実際に事故や規制などが行われているにもかかわらず判別に関する研究は行われていない。本研究ではその「ながら走り」をアンドロイドの制御下にある加速度センサとGPSを用いた判別法を提案する。加速度センサより得られる数値を元に判別用の値を求め、その値を時間当たりで平均化をしその数値が閾値の範囲にある場合はある程度の割合で「ながら走り」を検出することができる。精度を高めるためにGPSで速度計測した数値と閾値を比較し、範囲内にある場合は自転車であるとし、組み合わせることで精度を上げた。その後平均化した数値とGPSの数値を元に計測開始時より30秒以内に1秒ごとにデータの取得を行い、両方の数値が閾値に収まっていれば「ながら走り」と判別するものとした。本手法の評価を行ったところ、判別率は9割という結果が得た。

目次

第1章	はじめに	1
1.1	研究背景と目的	1
1.2	論文構成	5
第2章	提案手法と実装	6
2.1	アンドロイドOSと対応センサー一覧	6
2.2	加速度による振動の計測	7
2.3	GPSを用いた速度計測	10
2.4	状態の判別	12
第3章	検証と考察	14
3.1	検証環境	14
3.2	検証データ	15
3.3	本手法の検証結果	15
3.4	他の状態での本手法の検出結果	17
3.5	検出速度の検証	19
3.6	考察	19
第4章	まとめ	20
	謝辞	21
	参考文献	22

目 次

1.1	携帯電話販売台数予測と推移 ©総務省, 2009	2
1.2	歩きスマホに対する広告	3
1.3	ドコモ 警告画面	4
2.1	アンドロイド機器の加速度センサ	7
2.2	自転車の GPS 時速グラフ	10
2.3	歩きの GPS 時速グラフ	11
2.4	バスの GPS 時速グラフ	11
2.5	電車の GPS 時速グラフ	11
2.6	検出までのフローチャート	13
3.1	dell streak 001DL	15
3.2	自転車の時間ごとの GPS 速度	16
3.3	歩きの時間ごとの GPS 速度	17
3.4	電車の時間ごとの GPS 速度	18
3.5	バスの時間ごとの GPS 速度	18

第 1 章

はじめに

1.1 研究背景と目的

近年、スマートフォンの普及によりスマートフォンのユーザが増加している。図 1.1 は世界の携帯電話販売台数にしめるスマートフォンの販売台数の予測と推移である [1]。

これはパソコンのように様々なアプリケーションをユーザの好みで自由にインストールしてカスタマイズが可能であることや、通信速度の大きな発達により様々な情報をリアルタイムで発信、受信することが可能になったためである。しかしながら一方でマナーの悪いユーザによるスマートフォンによる事故事件が多発している現状もある。これはスマートフォンから取得できる情報量が多くスマートフォンに集中することが多くなったからだと考えられる。近年では、スマートフォンを使用しながら歩く「ながら歩き」による事故が問題となっている。東京都ではこのような事件が実際に起き、日本経済新聞 2013 年 11 月 06 日電子版 [2] によれば徒歩や自転車で移動中にスマートフォンや携帯電話を操作したことが牽引となった事故で消防庁が今年 1 月から 9 月までで計 24 人を救急搬送していることが 5 日報道されている。他にも駅のホームで「ながら歩き」をし、駅のホームから転落する事故や接触事故などが年々増加している [3]。国交省のまとめ [4] によると、ホームから転落したり、電車に接触したりして乗客が死傷する事故は、今年 4 月からの半年間に全国で 119 件発生している。これは、統計を取り始めた 02 年以降、

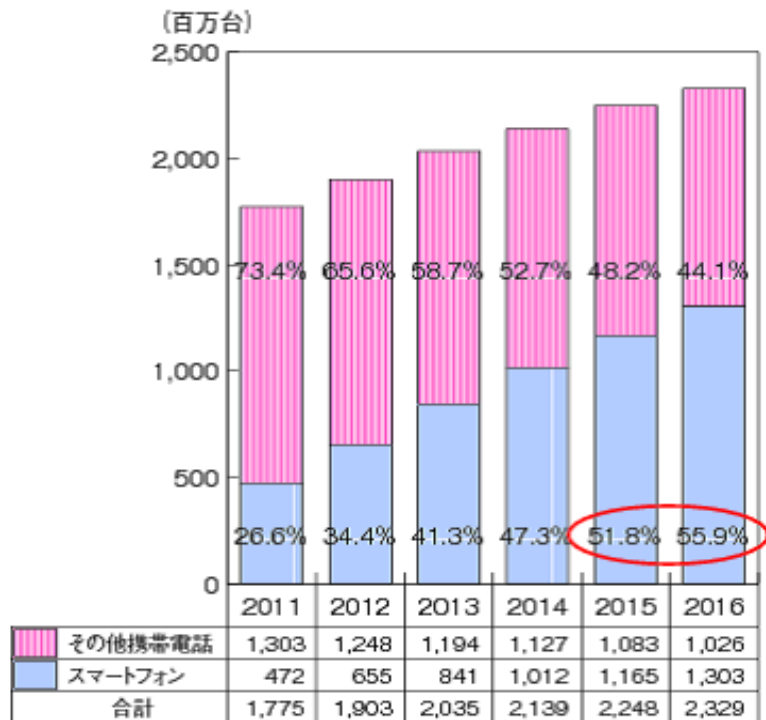


図 1.1: 携帯電話販売台数予測と推移 ©総務省, 2009

最悪のペースでの増加という。6割は飲酒による転落事故であるが、最近普及し始めたスマートフォンや携帯ゲームの操作に気をとられて起きる事故の増加も懸念されている。図 1.2 は新宿駅構内の「ながら歩き」に対する警告である。

新宿駅の構内では「ながら歩き」危険性に対する警告が出されている [5]。海外でも同じように「ながら歩き」は問題となっている。そのため「ながら歩き」に対して対策がとられている。情報通信総合研究所 infoCom ニュースレター [6] によればインドでは音楽をききながらでながら歩きをすることに対しての罰金が制定されている。更にアメリカ・ニュージャージー州のフォートリーでは、歩きながらスマホでメールする行為を禁じる「歩きスマホ規制条例」が 2012 年成立した。違反者には 85 ドルの罰金が科せられる [7]。以上に述べたようなことから「ながら歩き」はとても危険であることがわかる。そのためスマートフォン自体がその「ながら歩き」を検出することで事故を未然に防ぎ問題を解決するために検出手法を提案する研究が行われている。その「ながら歩き」を問題視し検出する研究とし



図 1.2: 歩きスマホに対する広告

て、加速度センサのデータを決定木による推移と状況を限定することで検出率を上げて「ながら歩き」を検出する澤野ら [8] のスマートフォン搭載センサを用いた「ながら歩き」検出方式や、カメラ画像をオプティカルフロー推定法の手法を利用してユーザの移動状況を検出する名坂ら [9] のスマートフォン使用時の不注意における事故防止システムの提案などがある。そして 2013 年にはドコモ [10] が「ながら歩き」に対する新たな取り組みを行った。ドコモの安心モードに歩きスマホ防止機能としてスマートフォンを使用しながら歩くと画面に警告が表示され、その間はスマートフォンの操作が不可能となる仕様である。図 1.3 はドコモの歩きスマホ防止機能が作動した時の画面の図である。全てに共通する検出の方針として「ながら歩き」はスマートフォンにかかる振動が一つの重要な要素である。

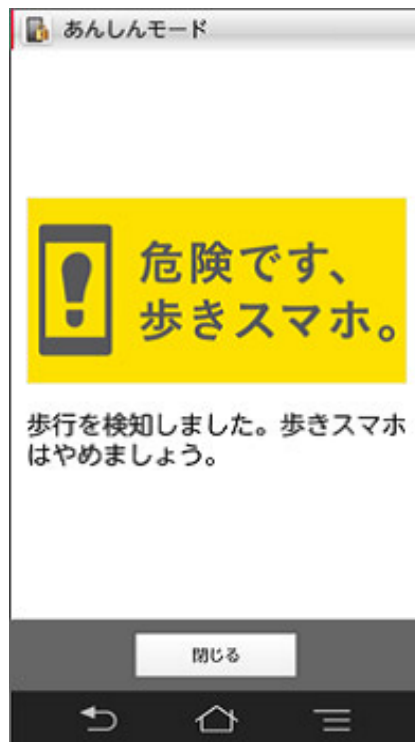


図 1.3: ドコモ 警告画面

同じように「ながら走り」も危険性がある。2013年11月20日の毎日新聞 [11] に「走りスマホ：やめて 徳島県警が取り締まり強化 苦情や県議会の要望受け」という報道があった。重大事故につながる前に防止しようと、自転車に乗りながらスマートフォン（スマホ）や携帯電話を使う「走りスマホ」に対する取り締まり強化に徳島県警が乗り出した。しかしながらスマートフォンを使用しながら自転車に乗り危険な運転をする「ながら走り」の検出に関する研究は少ない。センサを利用してユーザの状態を検出する関連研究としてはユーザの状態を検出する研究として進藤ら [12]、北川ら [13]、米村ら [14] などが存在する。また、小林ら [15] の研究ではモバイル機器を想定したセンサを利用してユーザの様々な移動状態を検出する方式を提案している。これらの研究はユーザの一定の状態を検出を目的としているため「ながら走り」の状態を取得できる研究報告はない。小林ら [15] の研究では、加速度センサ、マイク、GPS のモバイル機器に搭載されている機能を用いて取得したデータをパワースペクトルで解析することにより移動状態を推定

することが出来る方式を提案しており、その中には自転車に乗っている状態を検出するものも考慮されている。しかしながらこの研究では複数の状態を検出するために一定時間のデータの取得が必要となる。ユーザが歩行状態であるか 30 秒間かけ推定し、その次に 180 秒かけて自転車に乗っているかの状態を推定するためリアルタイムに自転車に乗っていてながら走りをしているかを検出するには推定時間に問題がある。検出時間を短縮するため、本研究ではアルゴリズムをシンプルに、複数のセンサを組み合わせることによってながら走りのみの検出の精度を高める方式を提案する。本研究の目的は、スマートフォン利用者が自転車に乗って運転している状態をリアルタイムに精度よく検出することにある。

本研究では関連研究の手法を参考に加速度センサと GPS を利用した「ながら走り」の状態を取得する手法を新しく提案する。加速度センサから得られる数値を元に平面ベクトルとし、その長さを計測開始時の時間から計測中までの時間で平均化した。その数値を閾値と比較し、更に GPS から取得できる速度を閾値と比較することで「ながら走り」の状態を取得することができた。「ながら走り」以外の「ながら歩き」、電車に乗っている状態、バスに乗っている状態で本手法を行ったところ「ながら走り」の状態は取得することはなかった。以上のことから提案手法の有用性を確認した。

1.2 論文構成

本論文は、本章を含め全 4 章からなり、第 2 章では、第 1 章で挙げている手法を述べ、第 3 章では検証と考察を行う。第 4 章では本研究のまとめを述べる。

第 2 章

提案手法と実装

本章では、本研究で提案するスマートフォンを使用しながら自転車を運転する「ながら走り」の検出手法について述べる。まず 2.1 節で手法を行うにあたっての想定環境を述べる。2.2 節ではアンドロイド機器に標準的に搭載されている加速度センサを利用したユーザの状態の取得について述べる。2.3 節では GPS を用いたユーザの状態の取得について述べる。2.4 節では 2.2 節、2.3 節で取得したデータを元にユーザがながら走りを行っているかの検出を行う手法について述べる。

本手法はアンドロイド機器とその制御下にあるセンサを使用してながら走りをしている状態を検出する物であり、アンドロイド機器の制御下にある加速度センサ、GPS センサとユーザが乗る自転車のみで構成する。使用する機材は、一般的な自転車とアンドロイド機器である。センサとして利用するのはアンドロイド機器の制御下にある加速度センサと GPS である。

2.1 アンドロイド OS と対応センサー一覧

Google より提供されるアンドロイド OS は現在 2013 年 12 月 19 日にはバージョン 4.4 まで存在している [16]。本手法ではながら走りを検出するため加速度センサと GPS を利用する。表 2.1 はアンドロイド OS のバージョンごとのセンサー対応表である。

表 2.1: バージョン対応のセンサー表

センサ名	単位	対応バージョン
加速度センサー	m/s^2	1.6
ジャイロセンサー	rad/s	1.6
照度センサー	lux	1.6
磁界センサー	μT	1.6
傾きセンサー	deg	1.6
圧力センサー	hPa	1.6
近接センサー	cm	1.6
温度センサー		1.6
重力センサー	m/s^2	2.3
直線加速度センサー	m/s^2	2.3
回転ベクトルセンサー	-	2.3
湿度センサー	パーセント	4.0

2.2 加速度による振動の計測

2.1 節で述べた表 2.1 より、アンドロイド機器より得られる加速度センサの数値を利用することで振動を数値化する。今回利用するものは加速度センサから取得できる数値である。アンドロイド機器搭載センサには機器の左側から受ける加速度 X と機器の下部から受ける加速度 Y、機器の裏面から受ける加速度 Z の 3 つが取得できる。図 2.1 はアンドロイド機器にかかる加速度を図に表わしたものである。

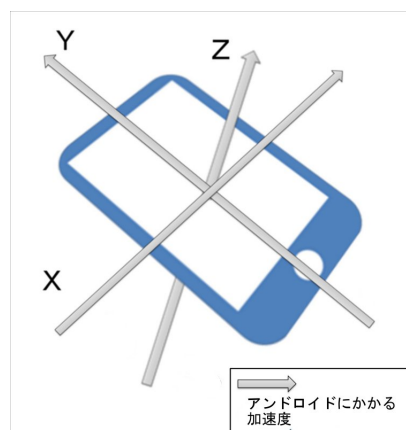


図 2.1: アンドロイド機器の加速度センサ

アンドロイド機器にかかる Y 方向の加速度の数値と Z 方向の加速度の数値を平面ベクトルとし、その長さを 1 秒ごとに平均化していく。本手法ではアンドロイド機器にかかる X 方向の加速度の数値は水平方向の回転のため使用はしない。初めにアンドロイド機器の加速度センサには重力加速度成分が含まれているためより正確な数値を取得するためローパスフィルタを用いて加速度から重力加速度をフィルタリングする必要がある。関連研究 [17][18] でも重力加速度の除去は行われている。今回使用するローパスフィルタは [19] に記述されているものを参考にした。加速度センサ Y より取得できる現在の加速度の数値を A_y 、 A_y より 1 つ前に取得した加速度の数値を B_y とする。同様に加速度センサ Z より取得できる現在の加速度の数値を A_z 、 A_z より 1 つ前に取得した加速度の数値を B_z とする。抽出したい重力加速度の数値を G_y 、 G_z とする。フィルタの数値は F とする。

$$\begin{cases} G_y = A_y F + B_y(1.0 - F) \\ G_z = A_z F + B_z(1.0 - F) \end{cases} \quad (2.1)$$

式 (2.1) より重力加速度成分をフィルタで抽出する。本手法では F の値を 0.1 とした。重力加速度をフィルタリングした数値を C_y 、 C_z とする。

$$\begin{cases} C_y = B_y - G_y \\ C_z = B_z - G_z \end{cases} \quad (2.2)$$

式 (2.1) より抽出した重力加速度成分を式 (2.2) にて減算することで重力加速度を除去した数値を求めることができる。次にフィルタリングした加速度の数値を元に 2 次元ベクトルの長さを求める。平面ベクトルの長さの数値を D とする。

$$D = \sqrt{C_y^2 + C_z^2} \quad (2.3)$$

式 (2.3) によって平面ベクトルの長さが分かる。取得開始時から現在までを平均化した数値 E を式 (2.4) によって求める。計測開始からの経過秒数 100 ミリ秒ごと

にデータを取得する。取得したデータの数を n とする。

$$E = \frac{1}{n} \sum_{i=1}^n D_i \quad (2.4)$$

取得したベクトルの大きさを元に式 (2.4) より計測開始時からの時間当たりの数値の平均化する。

表 2.2、表 3.4、表 2.4、表 2.5 はその計算式を元にデータを取得した表である。

表 3.4 より数値の幅は 0.79 から 1.0 となっている。そこから閾値としてバスの本手法数値の最大値 6.7 より大きい数値の下限 0.7 とし、上限を下限-0.1 に合わせた+0.1 の 1.1 の間に設定する。

表 2.2: 歩き本手法数値

試行回数	本手法数値
1 回目	1.2
2 回目	2.37
3 回目	2.86

表 2.3: 自転車本手法数値

試行回数	本手法数値
1 回目	0.8
2 回目	0.79
3 回目	0.8
4 回目	1.0

表 2.4: バス本手法数値

試行回数	本手法数値
1 回目	0.44
2 回目	0.67
3 回目	0.09

表 2.5: 電車本手法数値

試行回数	本手法数値
1 回目	0.34
2 回目	0.34

表 2.2、表 2.3、表 2.4、表 2.5 よりユーザの移動している状態ごとに値が一定の高さに収束していることが判明した。加速度センサのみで大まかな分類わけをすることができる。これにより自転車に乗っていて運転している状態と停止、歩き、電車に乗っている状態を大まかに区別することが出来る。

2.3 GPSを用いた速度計測

アンドロイド SDK[20] で使用できる GPS の機能は G3,WiFi を利用したネットワーク GPS と、衛星のデータを利用する GPS の 2 つがあり G3,Wifi を利用したネットワークの GPS では更新頻度が低いため精度は高くない。衛星の GPS ではアンドロイド SDK より、メートル毎秒の速度を出力が出来るため精度よく出力することが出来る。衛星の GPS は一度接続してしまえば更新頻度は高く、精度はある程度保障されている。そのため本手法では衛星の GPS を使用して速度を取得する。

次に自転車の平均走行速度は平均 15km/h である [21] とされている。そのためユーザがスマートフォンを操作し速度が落ちているという考慮に入れた上で実際にアンドロイド機器の GPS を使用して計測した数値を閾値に設定した。閾値は平均速度より遅い最少が 6km/h から最大が 15km/h の値とする。本研究での目標はスマートフォンを使用しながら自転車を運転している状態を取得することが目標である。そのため平地での自転車での走行中と停止状態と誤検出を行うかどうかを調べるため歩行、電車での移動、バスでの状態で本手法を利用してデータを取得した。図 2.2 図 2.3 図 2.4 図 2.5 は取得したデータをグラフにしたものである。

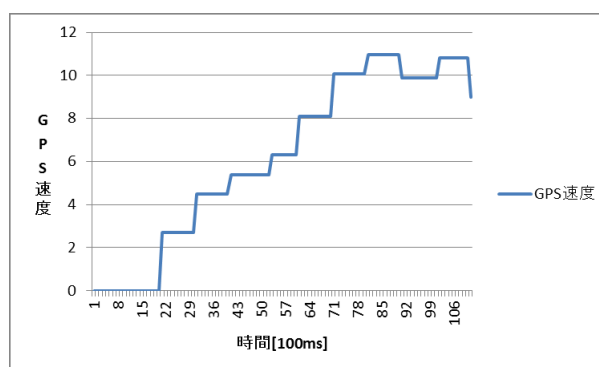


図 2.2: 自転車の GPS 時速グラフ

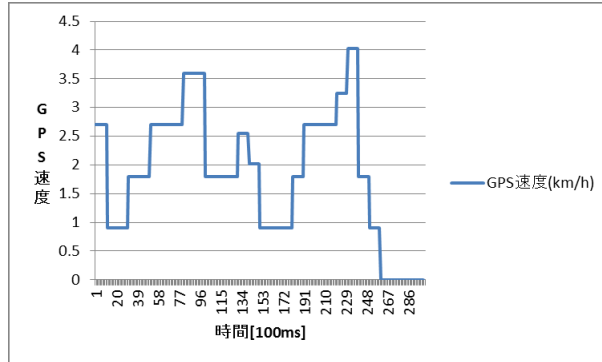


図 2.3: 歩きの GPS 時速グラフ

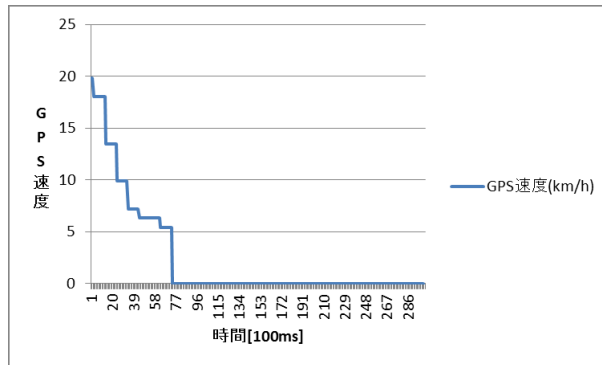


図 2.4: バスの GPS 時速グラフ

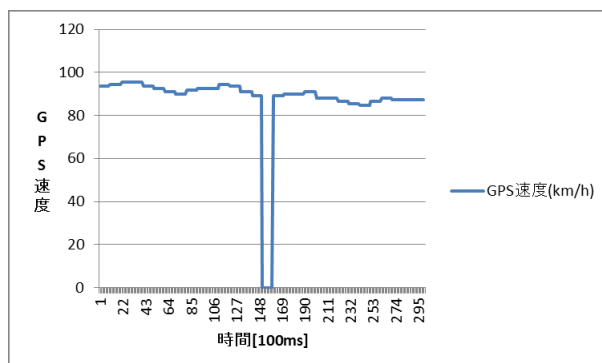


図 2.5: 電車の GPS 時速グラフ

2.4 状態の判別

2.2 節、2.3 節にて取得したデータを元にながら走りの検出するための手法を示す。アンドロイド機器の搭載センサより取得した数値を 100 ミリ秒ごとにデータとして保持する。その後 1 秒ごとに 2.2 節で取得したデータを元に 2.2 節の手法で数値を設定する。設定した数値を 2.2 節にて設定した閾値を比較し指定した数値に収まっているかを比較する。並行して 2.3 節にて述べた GPS を利用し計測した速度を取得し、最高速度の数値を更新する。そして最高速度が 2.3 節にて設定した閾値に収まっている場合はながら走りをしているとみなしアンドロイド機器にながら走りの状態を判別したことを送り処理を終了する。ながら走りをしているかを判別できずに指定時間を経過した場合、処理を終了とする。図 2.6 で大まかな流れを示す。

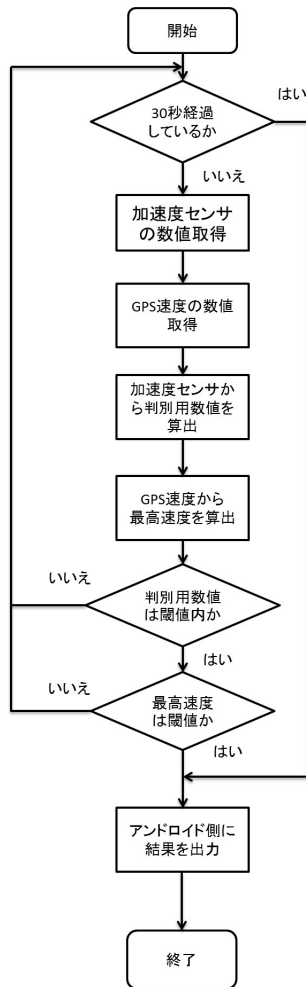


図 2.6: 検出までのフローチャート

第 3 章

検証と考察

本章では第 2 章で提案した手法を用いたプログラムの検証と考察を述べる。実装したプログラムは 2.1 節で述べたアンドロイド機器に対応しているソフトウェア開発キットであるアンドロイド SDK を用いて Java で開発をした。検証用アプリは検証開始し 30 秒以内にながら走りの状態を取得できなければ検証を中止し終了する。研究の目的である自転車と、そのほかのユーザの状態を制作したプログラムで検証した。

3.1 検証環境

検証環境で使用するアンドロイドスマートフォンは図 3.1 の dell streak 001DL、アンドロイド OS のバージョンは 2.2.2、その他仕様詳細は表 3.1 である。

表 3.1: dell streak 仕様詳細

OS 種類	Android 2.2.2
内蔵メモリ	ROM:2.5GB RAM:512MB
CPU	Qualcomm QSD8250 1GHz



図 3.1: dell streak 001DL

3.2 検証データ

男性 5 名に対して本手法を用いたアプリケーションを利用してながら歩き、ながら走りを対象としてながら走りの状態が取得できるか検証を行った。更にバス、電車についても男性 1 名のデータを取得した。

3.3 本手法の検証結果

本節では、本手法の実行時の計測回数とながら走り状態の取得回数を示す。本手法の検証結果は表 3.2 のとおりとなった。

表 3.2: 試行回数

	試行回数	取得回数
歩き	7	0
自転車	9	8
バス	3	0
電車	4	0

表 3.3: 自転車での試行でのながら走り取得秒数

	取得時の秒数
1 回目	11sec
2 回目	13sec
3 回目	19sec
4 回目	10sec
5 回目	2sec
6 回目	6sec
7 回目	5sec
8 回目	4sec
9 回目	15sec

表 3.4: 自転車での試行時の本手法数値

試行回数	本手法数値
1 回目	0.68
2 回目	0.96
3 回目	0.76
4 回目	0.77
5 回目	0.84
6 回目	0.8
7 回目	0.79
8 回目	0.8
9 回目	1.0

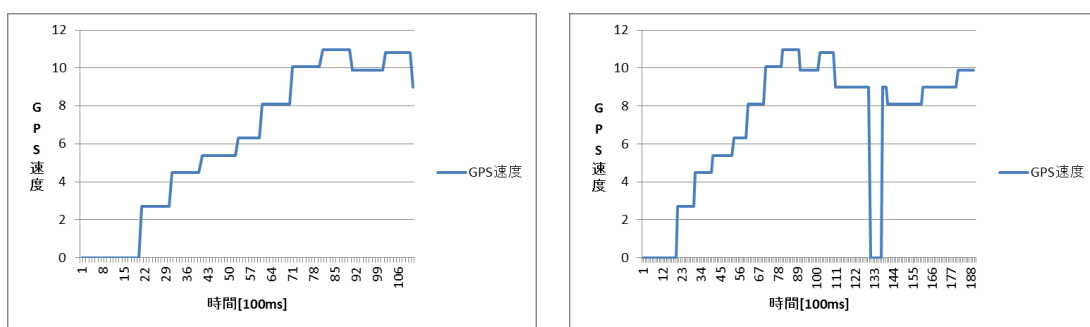


図 3.2: 自転車の時間ごとの GPS 速度

表 3.3 より自転車に乗っている状態の取得に成功している。表 3.4 は表 3.3 の 9 回分の本手法で計算した数値である。2.2 節にて述べた本手法の閾値以内にほぼ 9 割収まっている。1 回のみ誤検出として閾値を下回る結果となってしまった。更に GPS 速度も閾値に収まっており、問題なくながら走りを取得することが出来た。

3.4 他の状態での本手法の検出結果

本手法を用いた他の状態での検出結果の表 3.5, 表 3.6, 表 3.7 を示す。表 3.5 は本手法を歩きで試したものである。表 3.7 は本手法を電車に乗っている状態で試したものである。表 3.6 は本手法をバスに乗っている状態で試したものである。GPS の速度は自転車の速度とは違い、本手法での閾値からも離れているのでながら走りの状態を取得するにはいたらなかった。そのため 3 つの状態において本手法では誤検出は起こらなかった。

表 3.5: 歩きでの試行時の本手法数値

試行回数	本手法数値
1 回目	1.17
2 回目	2.36
3 回目	2.84

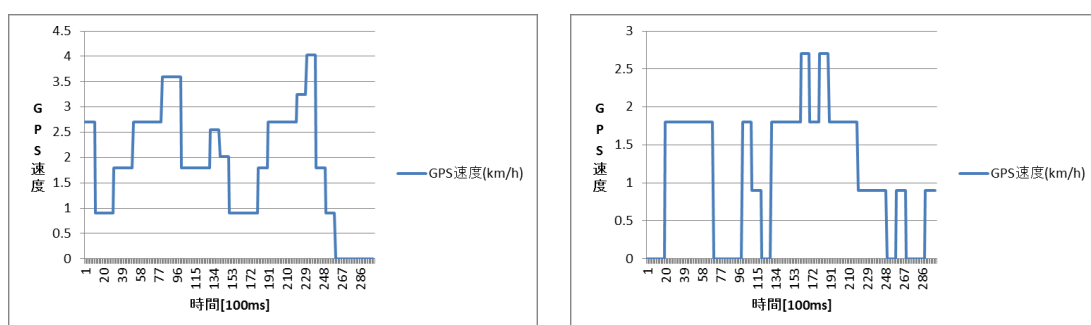


図 3.3: 歩きの時間ごとの GPS 速度

表 3.6: 電車での試行時の本手法数値

試行回数	本手法数値
1回目	0.34
2回目	0.34
3回目	0.27

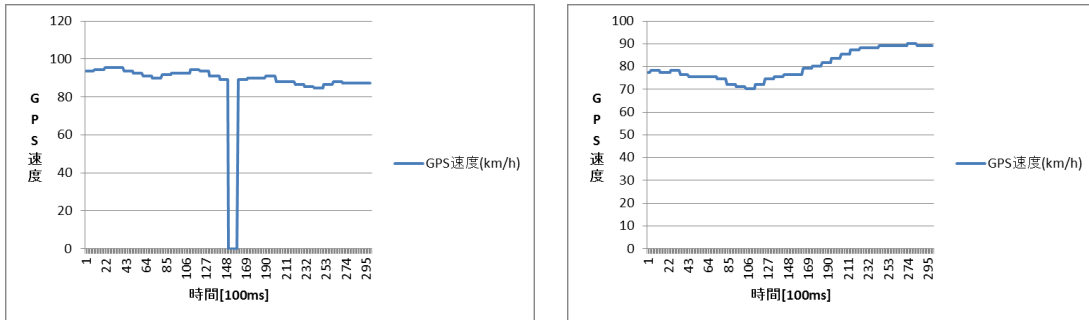


図 3.4: 電車の時間ごとの GPS 速度

表 3.7: バスでの試行時の本手法数値

試行回数	本手法数値
1回目	0.44
2回目	0.67
3回目	0.09

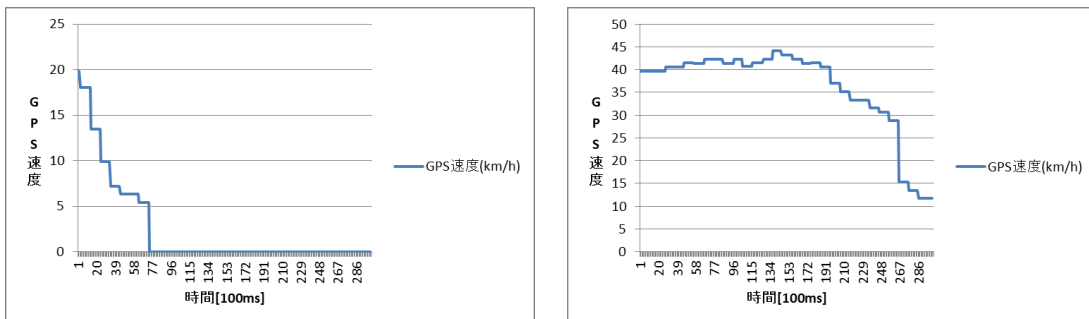


図 3.5: バスの時間ごとの GPS 速度

3.5 検出速度の検証

表 3.3 より数値のばらつきはあるが 20 秒を下回るという結果になった。垂令らの先行研究 [15] の大幅な時間をかけて状態を取得するものと比較しても非常に早い状態の検出率が出る結果となった。

3.6 考察

検証の結果、提案した手法により、非常に高い精度で状態の取得に成功することができた。更にユーザがほかの移動状態の時はながら走りの取得が起こらず、自転車でながら走りの時のみ状態の取得をすることができ、また検出までの時間についても 20 秒以下で取得することができた。しかし現状の問題点として GPS に接続しなければ最終的な処理の決定が行えないためそこが改善すべき点である。もう 1 つの問題点として GPS に接続し、高速で取得を繰り返すため電力の消費による使用時間の減少も問題である。

第 4 章

まとめ

本研究では、アンドロイドの搭載センサのみを利用してユーザが自転車に乗っている状態、「ながら走り」を検出する手法を提案した。この手法とはアンドロイドに標準的に搭載されている加速度センサ、GPS を利用した検出システムである。これによりユーザが「ながら走り」を行っていることを9割検出することができる結果を得られた。更に歩き、電車、バスに乗っている状態を誤検出することもないという結果も得られた。一方で検出頻度の問題からアンドロイド側の電池消費の問題や、GPS の届かない場所では検出が出来ないという問題点も挙げられる。これらを改善することにより、より「ながら走り」の検出の精度が上がるだろう。

謝辞

本研究を締めくくるにあたり、研究生活を見守ってくださり、研究方針、論文執筆等多大なお指導とご協力をいただきました多くの方への感謝の意を表します。

参考文献

- [1] 総務省. スマートフォン等の急速な普及と端末市場の変化. <http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h24/html/nc122110.html>.
- [2] 日本経済新聞. 日本経済新聞 2013/11/06 電子版. http://www.nikkei.com/article/DGXNASDG0503Y_V01C13A1CC1000/.
- [3] JCAST ニュース. スマートフォン「ながら歩き」は危険 高齢者や子ども、視覚障害者が被害. <http://www.j-cast.com/2011/10/14110061.html?p=all>.
- [4] 日テレ news24. 駅ホーム“ながら歩き”で死傷事故増加. <http://www.news24.jp/articles/2011/12/05/07195796.html>.
- [5] はてなブックマークニュース. 「危険です、歩きスマホ（本人は、この広告見ないだろうけど）」 ドコモ、新宿駅にマナー広告. <http://b.hatena.ne.jp/articles/201308/15458>.
- [6] 情報通信総合研究所 infoCom ニュースレター. 「ながら歩き」は危険がいっぱい. http://www.icr.co.jp/newsletter/global_perspective/2013/Gpre201365.html.
- [7] NEWS ポストセブン. 歩きスマホ規制条例 米ニュージャージー州では罰金 85 ドル. http://www.news-postseven.com/archives/20130428_184515.html.

- [8] 澤野 敬斗, 秋穂 崇, 新津 善弘. スマートフォン搭載センサを用いた「ながら歩き」検出方式. 電子情報通信学会, Vol. 11, p. 124, 2013.
- [9] 名坂 康平, 加藤 岳久, 西垣 正勝. スマートフォン使用時の不注意による事故防止システムの提案. 情報処理学会, Vol. Vol.2012-DPS-150 No.28, pp. 1-6, 2012.
- [10] NTT DOCOMO. 歩きスマホ防止の新たな取り組みについて. https://www.nttdocomo.co.jp/info/news_release/2013/12/03_00.html#p01.
- [11] 毎日新聞. 走りスマホ：やめて 徳島県警が取り締まり強化 苦情や県議会の要望受け. <http://sp.mainichi.jp/select/news/20131120mog00m040040000c.html>.
- [12] 進藤 康孝, 鏡 慎吾, 橋本 浩一. 人物の姿勢情報を用いた加速度・角速度センサの所持位置と向きへの推定. 電子情報通信学会技術研究報告. UBI, ユビキタス・センサネットワーク, [ユビキタスコンピューティングシステム] 2013-UBI-38(4), pp. 1-6, 2013-05-09.
- [13] 北川 拓, 新井 イスマイル. スマートフォン内蔵ジャイロセンサによる屋内方位推定精度向上. 情報処理学会研究報告. UBI, [ユビキタスコンピューティングシステム], [ユビキタスコンピューティングシステム] 2013-UBI-37(11), pp. 1-8, 2013-03-07.
- [14] 米村 淳, 大岸 智彦, 井戸上 彰, 小花 貞夫. スマートフォンを用いた人の混雑度推定手法の提案と評価. 情報処理学会研究報告. MBL, [モバイルコンピューティングとユビキタス通信研究会研究報告] 2013-MBL-67(5), pp. 1-8, 2013-09-05.
- [15] 小林 亜令, 岩本 健嗣, 西山 智. 釈迦：携帯電話を用いたユーザ移動状態推定・共有方式. 電子情報通信学会技術研究報告. MoMuC, モバイルマルチメディア通信 108(44), pp. 115-120, 2008.

- [16] google. android-developersblog. <http://android-developers.blogspot.jp/2013/10/android-44-kitkat-and-updated-developer.html>.
- [17] 長 健太, 池谷 直紀, 岡本 雄三, 瀬戸口 久雄, 服部 正典. センサを用いたモバイル機器向けユーザ状況推定技術の実装と評価. 情報処理学会研究報告, Vol. Vol.2010-MBL-52 No.12, pp. 1-7, 2010.
- [18] 瀬戸口 久雄, 池谷 直紀, 長 健太, 服部 正典. モバイル機器の日常的な利用シーンを考慮した加速度ノイズ対応方式. 情報処理学会研究報告. EMB, 組み込みシステム 2010-EMB-16(56), pp. 1-7, 2010-03-19.
- [19] 江川 崇. Android SDK WG 第3回セッション. http://www.android-group.jp/index.php?plugin=attach&refer=%CA%D9%B6%AF%B2%F1%2FAndroid%20SDK%20WG%20C2%E83%B2F3%20%A5%BB%A5%C3%A5%B7%A5%E7%A5F3%20with%A5%D7%A5C1%CA%D9%B6%AF%B2%F1&openfile=sdk_wg_sensor_exam_egawa.pdf.
- [20] Android developers. Android developers. <http://developer.android.com/index.html>.
- [21] 諸田 恵士, 大脇 鉄也, 上坂 克巳. 我が国の自転車利用の実態把握. 土木技術資料, 土木技術資料 51-4, pp. 1-9, 2009.